IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Jones et al.**	§	
	§	Group Art Unit: **2154**
Serial No. **09/888,473**	§	
	§	Examiner: **Dustin Nguyen**
Filed: **June 25, 2001**	§	
	§	
For: **Method and Apparatus for Wide-Spread Distribution of Electronic Content in a Peer to Peer Fashion**	§	

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

35525
PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

APPEAL BRIEF (37 C.F.R. 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on November 28, 2006.

No fees are required for the filing of this Appeal Brief. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0447. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0447.

REAL PARTY IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines Corporation of Armonk, New York.

RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

STATUS OF CLAIMS

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

Claims in the application are: 1-19.

B. STATUS OF ALL THE CLAIMS IN APPLICATION

1. Claims canceled: 3 and 11.
2. Claims withdrawn from consideration but not canceled: none
3. Claims pending: 1, 2, 4-10 and 12-19.
4. Claims allowed: none.
5. Claims rejected: 1, 2, 4-10 and 12-19.
6. Claims objected to: none.

C. CLAIMS ON APPEAL

The claims on appeal are: 1, 2, 4-10 and 12-19.

STATUS OF AMENDMENTS

No amendment after final rejection was filed for this case.

SUMMARY OF CLAIMED SUBJECT MATTER

A. CLAIM 1 - INDEPENDENT

Claim 1 is generally directed to an improved technique for eliminating bottlenecks that arise when downloading content from a server. A peer-to-peer offloading technique is provided for offloading of demands on master servers to other clients which are downloading the same content.

Specifically, Claim 1 is directed to a method for distributing information in a computer network. An electronic file is *divided into a plurality of pieces*. All of these file pieces are downloaded to a plurality of client machines, where the client machines function as peer-to-peer servers for other client machines requesting the file pieces. Each of these peer-to-peer servers stores *a unique file piece of the plurality of file pieces which is not stored on other of the peer-to-peer servers*. A request for a file piece is received from a first client machine, and the requested file piece is downloaded to this first client machine. A request for this same file piece is received from a second client machine. If the file piece requested from the second client machine has previously been downloaded to the first client machine responsive to the request for the file piece from the first client machine, the request of the second client machine is redirected to the first client machine. Redirecting the request for a given file piece to the first client machine advantageously allows for redirecting work away from the machine that originally provided the file piece by instead sending the work request to the first client, such that the first client can satisfy the request of the second client machine (Specification page 10, line 11 – page 11, line 13; Figure 4, all steps). In addition, by processing requests for individual file pieces through such redirection, instead of processing requests for an entire file, these claimed features advantageously provide for scattering or seeding of file pieces across a peer-to-peer environment such that the file pieces can be more efficiently processed by eliminating potential network bottlenecks (Specification page 10, lines 1-22). The potential bottleneck elimination is provided since, instead of copying the entire file to multiple different servers was done in the past, each peer-to-peer server stores *a unique file piece of the plurality of file pieces which is not stored on other of the peer-to-peer servers*.

B. CLAIM 7 - INDEPENDENT

Claim 7 is directed to a method for distributing information in a computer network, including steps of (i) requesting one of a plurality of pieces of an electronic file by a first machine, wherein the electronic file is stored in a server, (ii) receiving the requested file piece from the server, (iii) receiving, by the first machine, a request for a file piece by a second machine, where the request is conditionally redirected from the server to the first machine based upon whether the server has previously provided the file piece to the first machine, and (iv) sending, by the first machine, the file piece to the second machine. This method is from the perspective of a first machine which is different from the server and the second machine, where this first machine requests and receives content (the requested file *piece*), and then receives a request for a file piece, *the request being redirected from the server which previously supplied the content* to the first machine. This second redirected request is then fulfilled by the first machine that previously requested and received the file piece. Thus, Claim 7 advantageously provides that *a machine that has previously requested and received content (a file piece) is itself able to satisfy a subsequent request for this same content*, where the request for the content has been redirected by the server which originally provided the content – thus offloading work (requests for a file piece) that is directed to the server (Specification page 10, line 11 – page 11, line 13; Figure 4, all steps).

C. CLAIM 8 - INDEPENDENT

Claim 8 is directed to a method for obtaining distributed information in a computer network. One of a plurality of pieces of an electronic file is requested by a first machine, where the electronic file is stored in a server. The requested file piece is received by the first machine, where the requested file piece is from a second machine that contains a copy of the file piece, the copy of the file piece on the second machine being the result of a previous request for the file piece from the second machine to the server and receipt of the file piece from the server to the second machine (Specification page 10, line 11 – page 11, line 13; Figure 4, all steps). This claim advantageously provides an offload of server workload, as even though a request is made for a part of a file that is stored on the server, the requested file piece is instead received from a *second machine that had previously requested and received the file piece from the server.*

D. CLAIM 9 – INDEPENDENT

Claim 9 is directed to a computer program product in a computer readable medium for use in a data processing system, for distributing information in a computer network. The computer program product includes instructions for dividing an electronic file into a plurality of file pieces. The computer program product also includes instructions for downloading all of the file pieces to a plurality of client machines, where the client machines function as peer-to-peer servers for other client machines requesting the file pieces. Each peer-to-peer server stores a unique file piece of the file pieces which is not stored on other of the peer-to-peer servers. The computer program product also includes instructions for receiving a request for a file piece from a first client machine, and instructions for downloading the requested file piece to the first client machine. The computer program product also includes instructions for receiving a request for the file piece from a second client machine, and instructions for redirecting the request of the second client machine to the first client machine if the file piece requested from the second client machine has previously been downloaded to the first client machine responsive to the request for the file piece from the first client machine (Specification page 10, line 11 – page 11, line 13; Figure 4, all steps; page 13, line 25 – page 14, line 11).

E. CLAIM 15 – INDEPENDENT

Claim 15 is directed to a computer program product for distributing information in a computer network, the computer program product comprising instructions for execution by a second client machine. The computer program product includes instructions for requesting one of a plurality of pieces of an electronic file from a server, where the electronic file is stored in the server, and instructions for receiving the requested file piece from the server. The computer program product also includes instructions for receiving a request for another file piece from a client machine, where the request for the file piece is conditionally redirected from the server to the second client machine based upon whether the server has previously provided the other file piece to the second client machine. The computer program product also includes instructions for sending the other file piece to the client machine (Specification page 10, line 11 – page 11, line 13; Figure 4, all steps; page 13, line 25 – page 14, line 11).

F. CLAIM 16 – INDEPENDENT

Claim 16 is directed to a computer program product for obtaining distributed information in a computer network, the computer program product comprising instructions for execution by a second client machine. The computer program product includes instructions for requesting one of a plurality of pieces of an electronic file from a server, where the electronic file is stored in the server. The computer program product also includes instructions for receiving, without further request of the requested file piece by the second machine, the requested file piece from a client machine containing a copy of the file piece in lieu of receiving the requested file piece from the server (Specification page 10, line 11 – page 11, line 13; Figure 4, all steps; page 13, line 25 – page 14, line 11).

G. CLAIM 17 – INDEPENDENT

Claim 17 is directed to a system for distributing information in a computer network, the system includes a dividing component which divides an electronic file into a plurality of file pieces. The system also includes a download component which downloads all of the file pieces to a plurality of client machines, where the client machines function as peer-to-peer servers for other client machines requesting the file pieces. Each peer-to-peer server stores a unique file piece of the file pieces which is not stored on other of the peer-to-peer servers. The system also includes a first receiver which receives a request for a file piece from a first client machine, a communications component which downloads the requested file piece to the first client machine and a second receiver which receives a request for the file piece from a second client machine. The system also includes a redirecting component which redirects the request of the second client machine to the first client machine if the file piece requested from the second client machine has previously been downloaded to the first client machine responsive to the request for the file piece from the first client machine (Specification page 10, line 11 – page 11, line 13; Figure 4, all steps; page 5, line 3 – page 9, line 22; Figures 1-3, all elements).

H. CLAIM 18 – INDEPENDENT

Claim 18 is directed to a system for distributing information in a computer network, the system includes a first component of a first machine which requests one of a plurality of pieces

of an electronic file from a server, where the electronic file is stored in the server; and a second component of the first machine which (i) receives the requested file piece from the server, (ii) receives a request for another file piece from a client machine, where the request for the another file piece is conditionally redirected from the server to the first machine based upon whether the server has previously provided the other file piece to the first machine, and
(iii) sends the other file piece to the client machine (Specification page 10, line 11 – page 11, line 13; Figure 4, all steps; page 5, line 3 – page 9, line 22; Figures 1-3, all elements).

I. CLAIM 19 – INDEPENDENT

Claim 19 is directed to a system for obtaining distributed information in a computer network, the system includes a communications component of a first machine requesting one of a plurality of pieces of an electronic file from a server, where the electronic file is stored in the server; and a receiver of the first machine which receives the requested file piece, without further request of the requested file piece by the first machine, from a second machine containing a copy of the file piece in lieu of receiving the requested file piece from the server. The copy of the file piece on the second machine is the result of a previous request for the file piece from the second machine to the server and receipt of the file piece from the server to the second machine (Specification page 10, line 11 – page 11, line 13; Figure 4, all steps; page 5, line 3 – page 9, line 22; Figures 1-3, all elements).

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The grounds of rejection to review on appeal are as follows:

1. Whether Claims 1, 2, 4-10 and 12-19 are over *Scott et al.* (US Patent Application No. 2002/0049760) in view of *Outten et al.* (US Patent No. 7,024,466) under 35 U.S.C. §103(a).

ARGUMENT

A. GROUND OF REJECTION 1 (Claims 1, 2, 4-10 and 12-19)

A.1. Claims 1, 2, 4, 9, 10, 12 and 17

Claim 1 recites steps of "dividing an electronic file into a plurality of file pieces" and "downloading all of said file pieces to a plurality of client machines, wherein the client machines function as peer-to-peer servers for other client machines requesting said file pieces, wherein each peer-to-peer server stores *a unique file piece of the plurality of file pieces which is not stored on other of the peer-to-peer servers*". As can be seen, an electronic file is divided into a plurality of file pieces, and different ones of these file pieces are downloaded to a plurality of client machines which function as peer-to-peer servers, where *each peer-to-peer server stores a unique file piece which is not stored on other of the peer-to-peer servers*. These claimed features advantageously provide for scattering or seeding of file pieces across a peer-to-peer environment such that the file pieces can be more efficiently processed by eliminating potential network bottlenecks (Specification page 10, lines 1-22). The potential bottleneck elimination is provided since, instead of copying the entire file to multiple different servers was done in the past (and as is taught by the cited *Scott* reference), each peer-to-peer server stores *a unique file piece of the plurality of file pieces which is not stored on other of the peer-to-peer servers*. In rejecting Claim 1, the Examiner states that the cited *Scott* reference teaches the claimed 'dividing' step at paragraphs 0055 and 0066 in that *Scott* describes file chunks. Applicants urge that *Scott* does not teach that a file is divided and downloaded as chunks to different servers, but rather that *an entire file is downloaded to multiple clients*, where these **entire duplicate copies of the entire file** can then be accessed to *read* individual chunks (paragraphs 0044, 0047 and 0058). Importantly, *Scott requires that duplicate copies of the **entire** file be maintained at each of the client devices*, as the files are located using a hashing scheme in which the *entire file is hashed* to obtain a hash ID value that is used to locate the entire file. For example, as described by *Scott* at paragraph 0051:

> *It is crucial that the file contents on the first fulfilling Peer2 (506) and the second fulfilling Peer3 (508) corresponding to HASH ID be identical in every respect*, since otherwise these parts of files may not fit together correctly and result in a

damaged or corrupted final file. This is why it is essential to pick a HASH function that will uniquely create a unique HASH code from a file's contents.

It is therefore urged that the features of Claim 1 – specifically the claimed feature of:

> each peer-to-peer server stores *a unique file piece of the plurality of file pieces which is not stored on other of the peer-to-peer servers*

- clearly differentiates the claimed dividing of an electronic file into a plurality of file pieces and uniquely storing individual file pieces on the servers such that each of these file pieces are not stored on the other servers from *Scott's description of maintaining entire copies of duplicated files at multiple client devices* - albeit which can then be accessed as chunks when reading the file (*Scott* page 4, paragraph 0044; page 5, paragraphs 0051-0055). Quite simply, since entire copies of files are stored at multiple different locations, there is no teaching or suggestion of the claimed *unique file pieces*. Thus, as there are missing claimed features that are not taught or suggested by the cited references, a proper *prima facie* case of obviousness has not been established with respect to Claim 1[1]. Thus, Claim 1 has been erroneously rejected[2].

In rejecting this unique file piece aspect of Claim 1, the Examiner states that *Scott* teaches that each peer-to-peer server stores a unique file piece at *Scott* Figure 6B, elements 17, 25, and 27 and paragraph 0041, 0049 and 0054, since the chunk size may statistically or dynamically be determined based upon parameter and/or number of peers which currently have the file size. Applicants respectfully submit that elements 17, 25 and 27 of Figure 6B are *file requests for reading data*, and are not directed to a *downloading operation that* ***uniquely stores*** *individual file pieces*, as claimed. The cited passage at *Scott* paragraph 0041 describes a central server that can be searched to determine what files are available and where they can be located, and this central server description is not directed to a *downloading operation that uniquely stores individual file pieces*, as claimed. The cited passage at *Scott* paragraph 0049 describes that in a

---

[1] To establish *prima facie* obviousness of a claimed invention, all of the claim limitations must be taught or suggested by the prior art. MPEP 2143.03. *See also, In re Royka*, 490 F.2d 580 (C.C.P.A. 1974) (emphasis added by Appellants).

[2] If the examiner fails to establish a *prima facie* case, the rejection is improper and will be overturned. *In re Fine*, 837 F.2d 1071, 1074, 5 USPQ2d 1596, 1598 (Fed. Cir. 1988).

timeout condition, a peer requesting *an entire file* will request the *entire file* from another peer when the timeout occurs, and this passage is not directed to a *downloading operation that* ***uniquely stores*** *individual file pieces*, as claimed. The cited passage at *Scott* paragraph 0054 describes *reading* a portion of a file and that the size of the portion read (i.e. the chunk) is variable, and this passage is not directed to a *downloading operation that* ***uniquely stores*** *individual file pieces*, as claimed. Quite simply, changing the chunk size of how much of a file *to read* (as per the teachings of the cited *Scott* reference) is a totally different operation from a download operation *that* ***uniquely*** *stores individual file pieces*, as claimed in Claim 1.

Still further with respect to Claim 1, *Scott requires* that the multiple copies of the files be exactly the same (and therefore they are not unique), such that a hashing function can be used to identify the files, notwithstanding that their file name may be different. As succinctly stated by *Scott* at paragraph 0051:

> [0051] It is *crucial* that the *file contents* on the first fulfilling *Peer2* (506) and the second fulfilling *Peer3* (508) corresponding to HASH ID **be identical in every respect**, since otherwise these parts of files may not fit together correctly and result in a damaged or corrupted final file. This is why it is essential to pick a HASH function that will uniquely create a unique HASH code from a file's contents.

Thus, it is urged that the Examiner failed to establish a *prima facie* showing of obviousness with respect to Claim 1, and therefore it is urged that Claim 1 is not obvious in view of the cited references as there are claimed features not taught or suggested by any of the cited references. In addition, a person of ordinary skill in the art would not have been motivated to modify such teachings due to *Scott* expressing the requirement that multiple identical copies of the entire files are maintained in the system such that they can be accessed using hashIDs[3].

[3] The fact that a prior art device could be modified so as to produce the claimed device is not a basis for an obviousness rejection unless the prior art suggested the desirability of such a modification. *In re Gordon*, 733 F.2d 900, 221 USPQ 1125 (Fed. Cir. 1984). Although a device may be capable of being modified to run the way [the patent applicant's] apparatus is claimed, there must be a suggestion or motivation *in the reference* to do so. *In re Mills*, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990).

A.2. Claim 5, 6, 13 and 14

Further with respect to Claim 5 (and dependent Claim 6), such claim recites that the digest is used to determine *file piece corruption* (as described in the Specification at page 12, line 10 – page 13, line 24). The *Scott* hash ID (which is alleged to teach the features of Claim 5) is used to *locate entire files*, and thus is different from the features of Claim 5 for at least two reasons. First, *Scott's* hash ID is with respect to an *entire file*, whereas the digest of Claim 5 is with respect to a *file piece* resulting from a file being divided into pieces (as per Claim 1). This entire file distinction can easily be seen in the actual passage cited by the Examiner in rejecting Claim 5. As described by *Scott* at paragraph 0032 (which is the exact passage used by the Examiner in rejecting Claim 5), *Scott* states:

> "[0032] Turning now to FIG. 2, the unique footprinting of a file 202 is illustrated in accordance with one specific embodiment of the present invention. In particular, the process by which a file 202 may be uniquely identified is through the application of HASH code 204. Each peer device 102a-102n that wishes to publish a file on the peer-to-peer network 100 first computes the HASH code 204 **of that file**. The HASH code 204 may be computed via conventional algorithms which generate a unique identifier based upon the content of a particular file. Examples of fingerprinting or hash code algorithm which may be used in conjunction with the technique of the present invention include, MD5 (described in RFC 1321, and attached hereto has Appendix A), and Keyed SHA1 (described in RFC 2841, and attached hereto has Appendix B). Each of these references is incorporated herein by reference in its entirety for all purposes" (emphasis added by Appellants).

This entire file (as taught by the cited reference) versus file piece (as claimed) distinction is critical when viewed in the context of a high performance peer-to-peer network as per the present invention, where reducing the size of units of information being transferred improves overall system performance by reducing bottlenecks (Specification page 9, line 29 – page 11, line 27). Secondly, using a hash ID *to locate a file*, as per the teachings of the cited reference (*Scott* page 4, paragraph 0047), is very different from *determining file integrity (corruption)*

using a digest, as per Claim 5. In rejecting Claim 5, the Examiner states that *Scott* teaches that if an error occurs during file transfer processing, resulting in a partial file transfer, the automated process may be configured to identify the portion of the desired file which were not received, and automatically select at least one different remote location for retrieving the remaining content (citing *Scott's* Figure 7 and paragraphs 0008 and 0054). Appellants respectfully submit that to the extent the cited reference determines an error condition, such error determination is not done *using a digest* for a given file piece, but instead the *Scott* error is determined by a time out condition (*Scott* page 5, paragraph 0050). Claim 5 expressly recites "sending *a digest for a given file piece* to each client machine which has received the given file piece" and "*determining whether said given file piece is corrupted using the digest*". It is thus urged Claim 5 is not obvious in view of the cited references, and it is further urged that the Examiner has failed to properly establish a *prima facie* showing of obviousness, as the Examiner has not established – or even alleged – that any of the cited references teach or otherwise suggest determining *whether a file piece is corrupted using a digest for the file piece*.

A.3. Claims 7, 15 and 18

With respect to Claim 7, such claim pertains to *redirecting of requests for individual file pieces*. The cited *Outten* reference teaches providing a user with a URL to another machine (edge server), and the content requested and provided is with respect to an *entire file* (*Outten* col. 6, lines 20-27). This distinction is critical when viewed in the context of a high performance peer-to-peer network as per the present invention, where reducing the size of units of information being transferred improves overall system performance by reducing bottlenecks (Specification page 9, line 29 – page 11, line 27). In addition, per Claim 7, the redirection of requests to the *first machine* is *conditioned upon* whether the requested file piece has previously been received by the first machine, *as requested by the first machine*. This claimed feature advantageously provides the seeding or distribution of file pieces, whereby a machine that has previously requested a file piece can itself fulfill a subsequent request for the file piece in that the subsequent request for the file piece is *redirected to the machine that itself had previously requested the file piece*. In contrast, per the teachings of the cited *Outten* reference, the requests for an *entire file are unconditionally processed* (*Outten* Claim 1, last element, where it states

"*wherein each main server is programmed or configured for directing recipient processors to edge servers to obtain requested content items*", with the content items being entire files, and not pieces of files as claimed (*Outten* col. 4, lines 29-37)). The teachings of the cited references are therefore different for at least three reasons, as will now be shown.

First, *Outten* processes requests for *entire* files (as contrasted with the claimed redirection of requests for file *pieces*). The entire premise of *Outten* is to provide a hierarchical environment of main servers, parent servers and edge servers in order to accommodate requests for very large files such as video (col. 2, lines 16-39), and thus a person of ordinary skill in the art would not have been motivated to modify such teachings in accordance with the request for file pieces feature of Claim 7, and thus this claimed feature is not obvious per *In re Gordon*, supra and *In re Mills*, supra.

Secondly, *Outten* redirection is unconditional (as contrasted with the claimed redirection of a request to a given machine which itself has previously requested and received the file piece), as previously described above.

Thirdly, the machine for which the request for a file piece is redirected to, per the features of Claim 7, is a machine that itself had previously requested and received the file piece ("the first machine"). In contrast, per the teachings of *Outten*, the machine (recipient processor) which is currently requesting a (entire) file is itself directed to another machine (edge server), by the main server providing to the machine (recipient processor) a URL of where to retrieve the content (i.e. the recipient processor machine is provided, by the main server, a URL to the edge server for content retrieval, as described by *Outten* col. 6, lines 54-62; col. 8, lines 53-63 and col. 9, lines 4-8). Quite simply, *Outten's* actual request itself is not redirected, as claimed in Claim 7, and *Outten's* redirection of content is not to a machine that itself had previously requested and received a file piece being requested, as claimed in Claim 7.

Thus, it is urged that Claim 7 is not obvious in view of the cited references as there are at least three claimed features not taught or suggested by any of the cited references.

A.4. Claim 8, 16 and 19

With respect to Claim 8, such claim recites "receiving, by the first machine and without further request of the requested file piece by the first machine, the requested file piece from a

second machine containing a copy of said file piece in lieu of receiving the requested file piece from the server, *the copy of said file piece on the second machine being the result of a previous request for the file piece from the second machine to the server and receipt of the file piece from the server to the second machine*". As can be seen, per the features of Claim 8, a first machine receives a file piece from a machine other than the one for which the request was directed (it is received from a second machine, whereas it was requested from a server), and the file piece that is received from the second machine is a copy of a file piece which was the result of *a previous request for this same file piece* by the second machine which is now providing the file piece to the first machine. None of the cited references teach or suggest *processing a request for a file piece differently* - providing the file piece from a machine (the second machine) other than the machine for which this request was directed (the server) - *depending upon whether the file piece has previously been requested*. In rejecting Claim 8, the Examiner relies on reasoning given in rejecting Claims 1, 2 and 7. Applicants urge that Claim 8 is different from Claims 1, 2 and 7, as described above, and mere reliance on the reasoning used in rejecting Claims 1, 2 and 7 does not establish a teaching or suggestion of the specific features recited in Claim 8 (as articulated above). Thus, a *prima facie* case of obviousness has not been established with respect to Claim 8, and therefore the burden has not shifted to Applicants to rebut such obviousness assertion[4]. In addition, as a proper *prima facie* showing of obviousness has not been established with respect to Claim 8, such claim has been erroneously rejected[5].

Further, and as described above with respect to Claim 7, the cited *Outten* reference is directed to processing entire files, whereas Claim 8 is directed to processing individual file piece requests. The entire premise of *Outten* is to provide a hierarchical environment of main servers, parent servers and edge servers in order to accommodate requests for very large files such as video (col. 2, lines 16-39), and thus a person of ordinary skill in the art would not have been

---

[4] In rejecting claims under 35 U.S.C. Section §103, the examiner bears the initial burden of presenting a prima facie case of obviousness. *In re Oetiker*, 977 F.2d 1443, 1445, 24 USPQ2d 1443, 1444 (Fed. Cir. 1992). Only if that burden is met, does the burden of coming forward with evidence or argument shift to the applicant. *Id.* To establish *prima facie* obviousness of a claimed invention, all of the claim limitations must be taught or suggested by the prior art. MPEP 2143.03. *See also, In re Royka*, 490 F.2d 580 (C.C.P.A. 1974).

[5] If the examiner fails to establish a *prima facie* case, the rejection is improper and will be overturned. *In re Fine*, 837 F.2d 1071, 1074, 5 USPQ2d 1596, 1598 (Fed. Cir. 1988).

motivated to modify such teachings in accordance with the request for file pieces feature of Claim 8, and thus this claimed feature is not obvious per *In re Gordon*, supra and *In re Mills*, supra.

Appellants have thus shown numerous and substantial error in the Examiner's final rejection of all pending claims, and respectfully request that the Board reverse such erroneous final rejection of all pending claims.

/Wayne P. Bailey/
Wayne P. Bailey
Reg. No. 34,289
YEE & ASSOCIATES, P.C.
PO Box 802333
Dallas, TX 75380
(972) 385-8777

CLAIMS APPENDIX

The text of the claims involved in the appeal are:

1. A method for distributing information in a computer network, the method comprising:

dividing an electronic file into a plurality of file pieces;

downloading all of said file pieces to a plurality of client machines, wherein the client machines function as peer-to-peer servers for other client machines requesting said file pieces, wherein each peer-to-peer server stores a unique file piece of said file pieces which is not stored on other of the peer-to-peer servers;

receiving a request for a file piece from a first client machine;

downloading the requested file piece to the first client machine;

receiving a request for said file piece from a second client machine; and

if said file piece requested from the second client machine has previously been downloaded to the first client machine responsive to the request for said file piece from the first client machine, redirecting the request of the second client machine to the first client machine.

2. The method according to claim 1, further comprising:

if said file piece requested from the second client machine has not previously been downloaded to the first client machine, processing the request for said file piece from the second client machine by a server which maintains a complete copy of the electronic file in lieu of redirecting the request of the second client machine to the first client machine.

4. The method according to claim 2, further comprising:

receiving a request for a file piece stored in a first peer-to-peer server which is no longer connected to the computer network;

redirecting said request to a second peer-to-peer server containing a copy of said file piece; and

removing the first peer-to-peer server from a list of available peer-to-peer servers.

5. The method according to claim 2, further comprising:

sending a digest for a given file piece to each client machine which has received the given file piece; and

determining whether said given file piece is corrupted using the digest.

6. The method according to claim 5, further comprising:

receiving a message from a client, wherein the message indicates that a peer-to-peer server has corrupted said given file piece;

disconnecting the peer-to-peer server responsible for corrupting said given file piece; and

retransmitting said given file piece to said client, wherein the retransmitted file piece is free of any corrupting content.

7. A method for distributing information in a computer network, the method comprising:

requesting, by a first machine, one of a plurality of pieces of an electronic file from a server, wherein the electronic file is stored in the server;

receiving, by the first machine, the requested file piece from the server;

receiving, by the first machine, a request for another file piece from a second machine, wherein the request for said another file piece is conditionally redirected from the server to the first machine based upon whether the server has previously provided said another file piece to the first machine; and then

sending, by the first machine, said another file piece to said second machine.

8. A method for obtaining distributed information in a computer network, the method comprising:

requesting, by a first machine, one of a plurality of pieces of an electronic file from a server, wherein the electronic file is stored in the server;

receiving, by the first machine and without further request of the requested file piece by the first machine, the requested file piece from a second machine containing a copy of said file piece in lieu of receiving the requested file piece from the server, the copy of said file piece on the second machine being the result of a previous request for the file piece from the second machine to the server and receipt of the file piece from the server to the second machine.

9. A computer program product in a computer readable medium for use in a data processing system, for distributing information in a computer network, the computer program product comprising:

instructions for dividing an electronic file into a plurality of file pieces;

instructions for downloading all of said file pieces to a plurality of client machines, wherein the client machines function as peer-to-peer servers for other client machines requesting

said file pieces, wherein each peer-to-peer server stores a unique file piece of said file pieces which is not stored on other of the peer-to-peer servers;

instructions for receiving a request for a file piece from a first client machine;

instructions for downloading the requested file piece to the first client machine;

instructions for receiving a request for said file piece from a second client machine; and

instructions for redirecting the request of the second client machine to the first client machine if said file piece requested from the second client machine has previously been downloaded to the first client machine responsive to the request for said file piece from the first client machine.

10. The computer program product according to claim 9, further comprising:

instructions for instructions for processing the request for said file piece from the second client machine by a server which maintains a complete copy of the electronic file, in lieu of redirecting the request of the second client machine to the first client machine, if said file piece requested from the second client machine has not previously been downloaded to the first client machine.

12. The computer program product according to claim 10, further comprising:

instructions for receiving a request for a file piece stored in a first peer-to-peer server which is no longer connected to the computer network;

instructions for redirecting said request to a second peer-to-peer server containing a copy of said file piece; and

instructions for removing the first peer-to-peer server from a list of available peer-to-peer servers.

13. The computer program product according to claim 10, further comprising:

instructions for sending a digest for a given file piece to each client machine which has received the given file piece; and

instructions for determining whether said given file piece is corrupted using the digest.

14. The computer program product according to claim 13, further comprising:

instructions for receiving a message from a client, wherein the message indicates that a peer-to-peer server has corrupted said given file piece;

instructions for disconnecting the peer-to-peer server responsible for corrupting said given file piece; and

instructions for retransmitting said given file piece to said client, wherein the retransmitted file piece is free of any corrupting content.

15. A computer program product for distributing information in a computer network, the computer program product comprising instructions for execution by a second client machine, including:

instructions for requesting one of a plurality of pieces of an electronic file from a server, wherein the electronic file is stored in the server;

instructions for receiving the requested file piece from the server;

instructions for receiving a request for another file piece from a client machine, wherein the request for said file piece is conditionally redirected from the server to the second client machine based upon whether the server has previously provided said another file piece to the second client machine; and

instructions for sending said another file piece to said client machine.

16. A computer program product for obtaining distributed information in a computer network, the computer program product comprising instructions for execution by a second client machine, including:

instructions for requesting one of a plurality of pieces of an electronic file from a server, wherein the electronic file is stored in the server;

instructions for receiving, without further request of the requested file piece by the second machine, the requested file piece from a client machine containing a copy of said file piece in lieu of receiving the requested file piece from the server.

17. A system for distributing information in a computer network, the system comprising:

a dividing component which divides an electronic file into a plurality of file pieces;

a download component which downloads all of said file pieces to a plurality of client machines, wherein the client machines function as peer-to-peer servers for other client machines requesting said file pieces, wherein each peer-to-peer server stores a unique file piece of said file pieces which is not stored on other of the peer-to-peer servers;

a first receiver which receives a request for a file piece from a first client machine;

a communications component which downloads the requested file piece to the first client machine;

a second receiver which receives a request for said file piece from a second client machine; and

a redirecting component which redirects the request of the second client machine to the first client machine if said file piece requested from the second client machine has previously been downloaded to the first client machine responsive to the request for said file piece from the first client machine.

18. A system for distributing information in a computer network, the system comprising:

a first component of a first machine which requests one of a plurality of pieces of an electronic file from a server, wherein the electronic file is stored in the server; and

a second component of the first machine which receives the requested file piece from the server, receives a request for another file piece from a client machine, wherein the request for said another file piece is conditionally redirected from the server to the first machine based upon whether the server has previously provided the another file piece to the first machine, and

sends said another file piece to said client machine.

19. A system for obtaining distributed information in a computer network, the system comprising:

a communications component of a first machine requesting one of a plurality of pieces of an electronic file from a server, wherein the electronic file is stored in the server;

a receiver of the first machine which receives the requested file piece, without further request of the requested file piece by the first machine, from a second machine containing a copy of said file piece in lieu of receiving the requested file piece from the server, the copy of said file piece on the second machine being the result of a previous request for the file piece from the second machine to the server and receipt of the file piece from the server to the second machine.

EVIDENCE APPENDIX

There is no evidence to be presented.

RELATED PROCEEDINGS APPENDIX

There are no related proceedings.